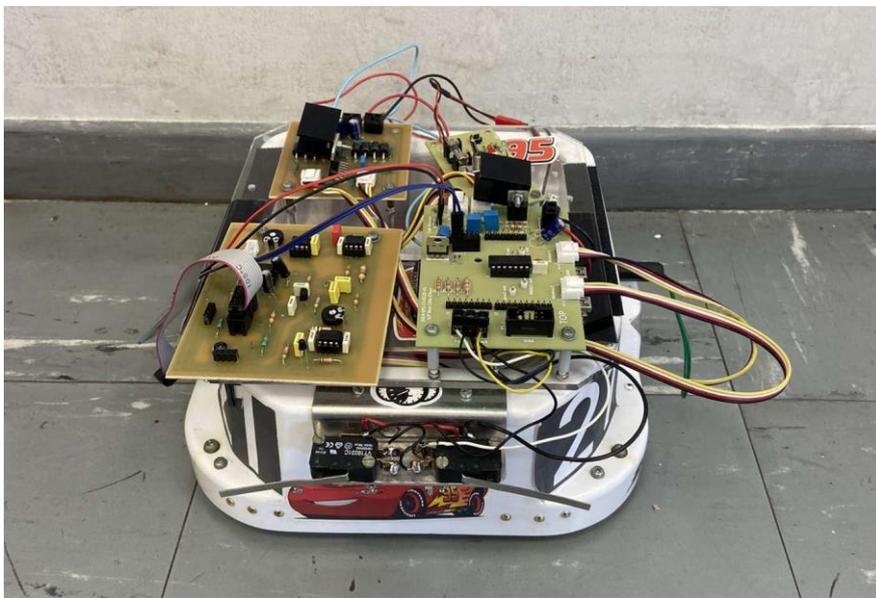


RAPPORT DE PROJET ROBOT

2022 – SEMESTRE 2



Groupe 36

Pichot Poncet Ermacora Rebuffel

Aline CABASSON

Christophe VERMAELEN

Sommaire

Début du projet.....	3
Organisation du projet.....	3
Diagramme de GANTT	4
Travail fait en cours	6
Télémètre SRF04	6
Chaîne de traitement du signal IR	8
Carte Mezzanine	18
a) Préparation.....	18
b) Conception du schématique	19
c) Conception du PCB.....	20
d) Impression et soudures	21
e) Problèmes rencontrés	22
Epreuves.....	23
a) Suivre une ligne blanche	23
b) Sortir du labyrinthe	23
c) Rejoindre la balise	24
Télécommande Bluetooth	24
Conclusion/Remarques	27
Annexes.....	28

Début du projet

Au début du projet, nous avons reçu notre robot et les objectifs à atteindre à la fin du projet. Les objectifs de ce semestre sont différents du semestre dernier. En effet, le cahier des charges est le suivant :

- Concevoir, souder puis tester les cartes capteurs et hacheur
- Suivre une ligne blanche.
- Détecter un obstacle à environ 25cm pendant le suivi de ligne et s'arrêter en conséquence.
- Sortir d'un labyrinthe.
- Détecter une balise infrarouge émettant une fréquence de 5kHz.
- Concevoir la carte mezzanine permettant de réaliser les trois épreuves précédentes et d'accueillir un connecteur Grove pour le LCD.
- Ajouter un module Bluetooth pour télécommander le robot.

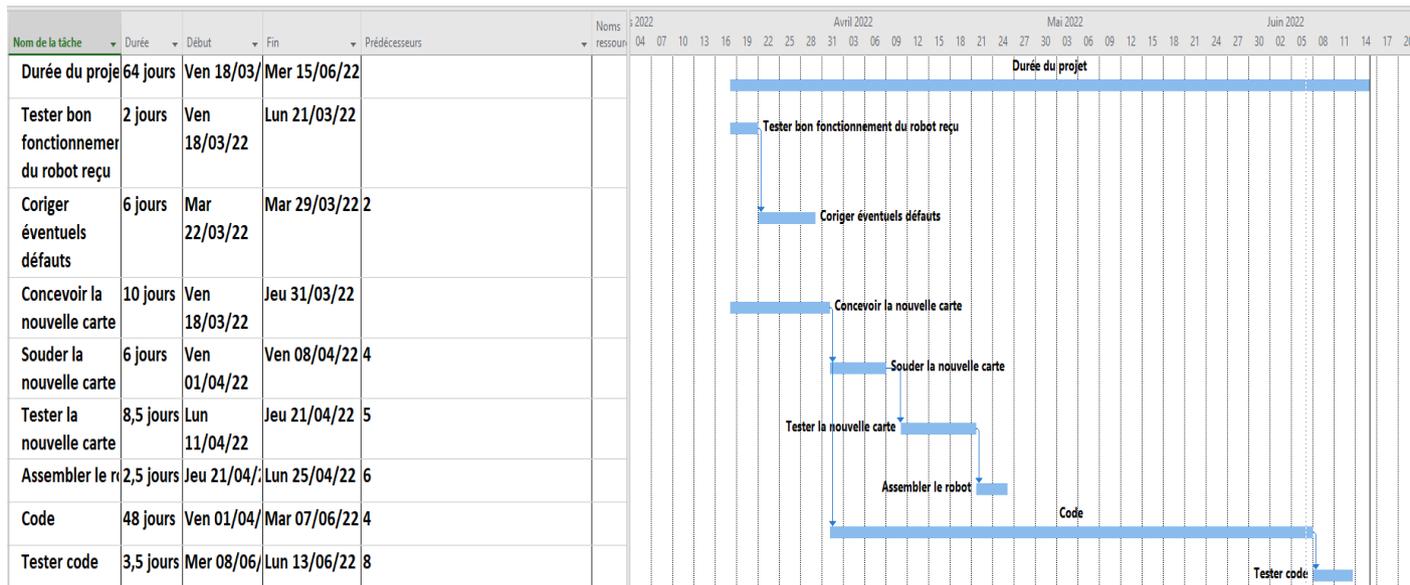
Le robot que nous avons reçu était complet et avait déjà en sa possession les cartes capteurs et hacheur fonctionnelles. C'est une chance qui nous a permis de gagner beaucoup de temps car nous avons pu nous concentrer dès le début sur l'étude et la conception de la carte mezzanine.

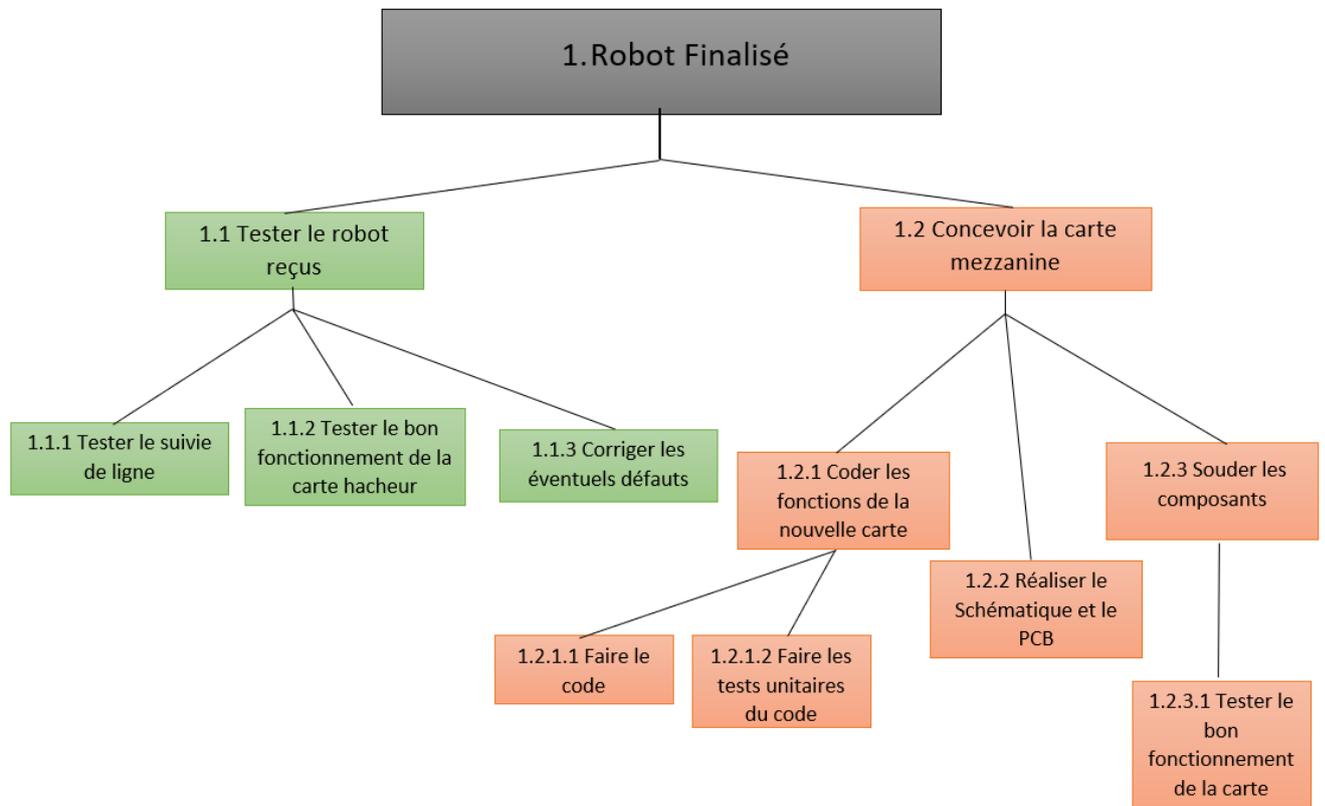
Organisation du projet

En vue des différentes tâches que nous avons à effectuer, nous avons en premier lieu organisé un diagramme de Gantt afin d'anticiper dans le temps nos objectifs. Nous nous sommes attribués beaucoup de jours de code car la programmation n'est pas notre point fort, par

conséquent nous devons passer plus de temps sur cette partie du projet.

Diagramme de GANTT





1. Robot Finalisé

1.1 Tester le robot reçus

1.1.1 Tester le suivie de ligne

1.1.2 Tester le bon fonctionnement de la carte hacheur

1.1.3 Corriger les éventuels défauts

1.2 Concevoir la carte mezzanine

1.2.1 Coder les fonctions de la nouvelle carte

1.2.1.1 Faire le code

1.2.1.2 Faire les tests unitaires du code

1.2.2 Réaliser le Schématique et le PCB

1.2.3 Souder les composants

1.2.3.1 Tester le bon fonctionnement de la carte

Travail fait en cours

En cours, nous devons étudier et concevoir une carte permettant d'effectuer les tâches suivantes :

- Détecter un obstacle
- Sortir du labyrinthe
- Détecter une balise infrarouge de 5kHz

Télémètre SRF04

Nous avons en premier lieu étudié le télémètre SRF04 qui nous permettrait à la fois de détecter un obstacle devant le robot et aussi sortir du labyrinthe.



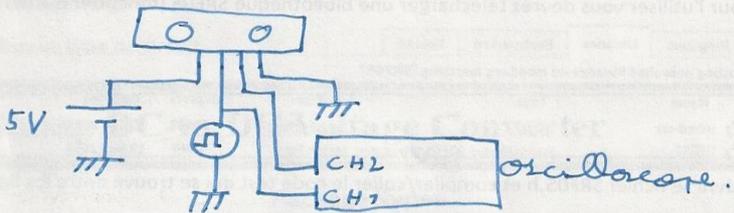
Photo du capteur SRF04

Le SRF04 est alimenté en 5v sur la première broche, reçoit un signal carré de 0-5V de 20Hz de fréquence avec un rapport cyclique de 1% sur la deuxième et renvoie un signal d'écho sur la troisième. La quatrième broche est reliée au GND. Plus l'obstacle est éloigné du capteur, plus le signal d'écho est grand (plus le ΔT est grand) Nous pouvons donc déterminer la distance. Dans le programme nous utiliserons la librairie adaptée à ce capteur.

c) Comment la forme de Echo change-t-elle en fonction de la distance entre le SRF04 et un objet ?

le Δt du temps haut de Echo augmente si la distance que mesure le capteur augmente.

→ Faire une fiche de mesure (utiliser différentes distances).

 <small>NICE CÔTE D'AZUR</small>	Fiche de mesure	
Module ou élément sous test télémètre SRF04		
Objectif du test comprendre et tester ce que renvoie le télémètre		
Schéma du banc de test (avec les symboles des appareils de mesure)		
		
Procédure de test (séquence des opérations) - Nous avons placé une planche tous les 20 cm et mesuré Δt $20 \text{ cm} = 7,25 \text{ ns}$ $40 \text{ cm} = 2,5 \text{ ns}$ $60 \text{ cm} = 3,75 \text{ ns}$		
Résultats de la mesure tous les 20 cm Δt augmente de 7,25 ns. l'augmentation de Δt et de la distance sont proportionnels		
Commentaire des mesures / Conclusion le télémètre fonctionne et Δt et distance (télémètre - objet) est proportionnel.		

Chaîne de traitement du signal IR

Une grande partie du travail réalisé en cours a été consacré au traitement du signal Infra Rouge de 5kHz. Pour cela, nous utilisons un phototransistor.

Cependant, celui-ci ne nous sert qu'à détecter un signal IR et non pas à déterminer sa fréquence.

Il y a en tout 6 étapes dans la chaîne de traitement :

1- **Capter le signal Infrarouge**

Nous avons donc le transistor alimenté en 5V et un AOp alimenté en +5V/-5V. Le transistor est branché sur l'entrée (-) de l'AOp. L'entrée (+) est à la masse. En sortie nous avons une contre réaction négative avec une résistance de 5kOhms.

Nous remarquons que plus le capteur est éloigné de la balise, plus la tension de sortie crête à crête est faible.

3. Chaîne de traitement du signal
 Bloc 1 « Capter »
 Bloc 3 « Amplifier »
 Bloc 5 « Récupérer l'enveloppe »
 Bloc 6 « Rendre le signal binaire »

Objectifs :

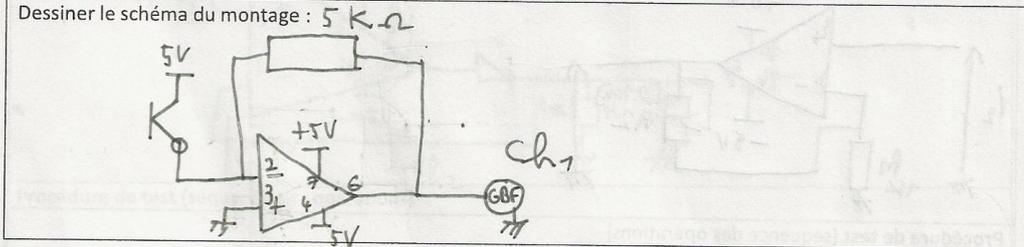
- Avoir compris la chaîne de traitement du signal
- Tester le capteur de balise IR
- Savoir amplifier un signal

Partie 7 - BLOC 1 - « Capter le signal IR »

Dans la vidéo de la chaîne de traitement du signal disponible sur LMS, le bloc numéro 1 est détaillé à partir de la minute 3'06. Sur labdec, vous devez refaire le montage, et le tester à l'aide de la balise disponible sur la paillasse.

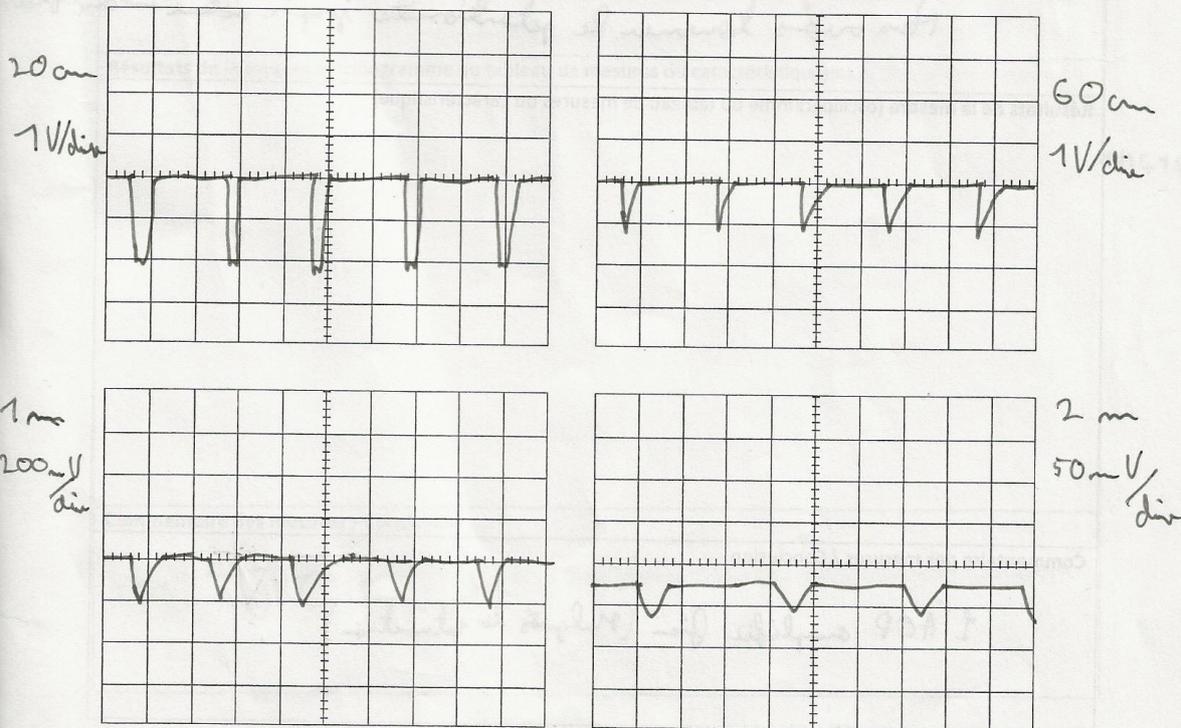
7.1 Schéma du détecteur IR

Dessiner le schéma du montage : $5\text{K}\Omega$

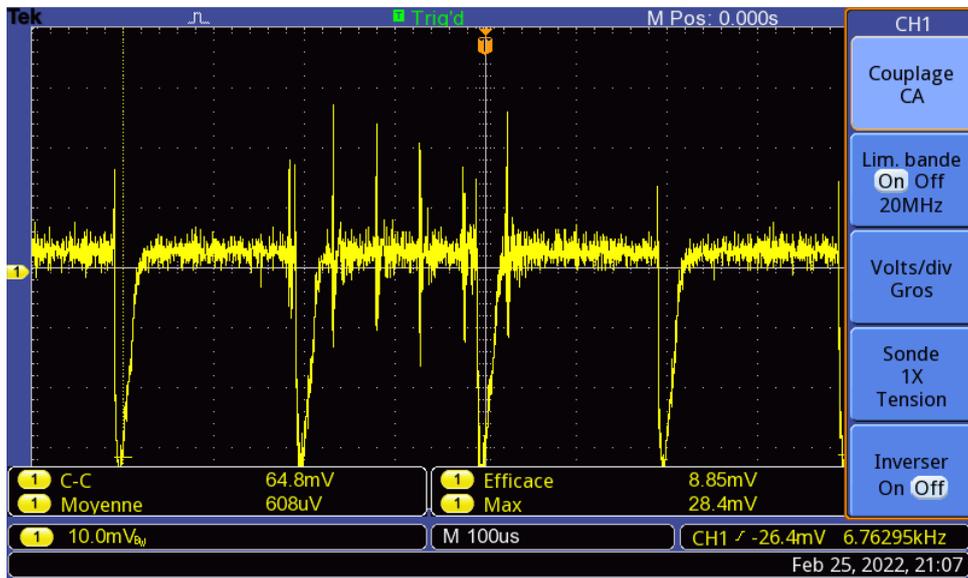


7.2 Test du détecteur IR

Pour une fréquence d'émission de 5kHz, relever la forme de la tension reçue pour 4 distances.

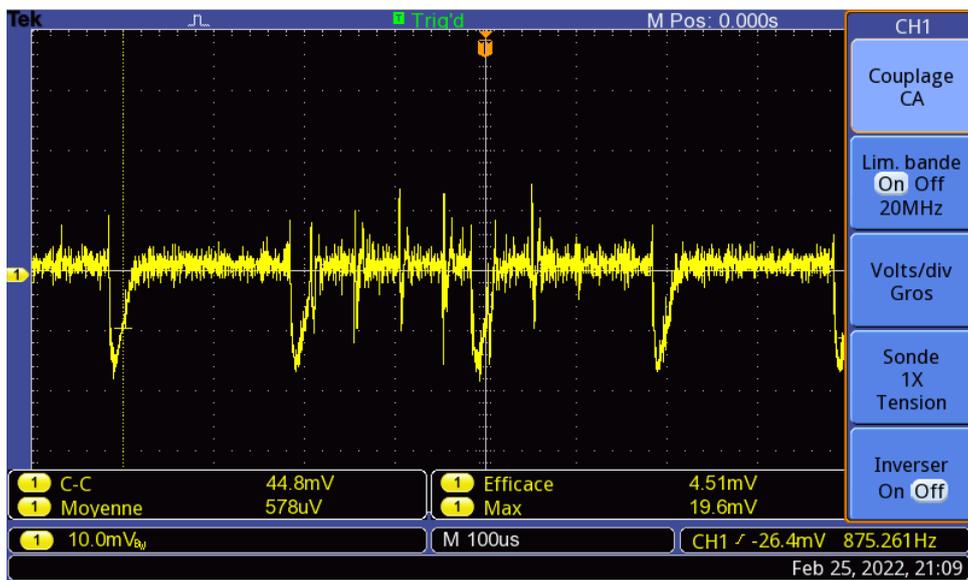


Montage pour capter le signal IR et les différents résultats selon la distance avec la balise



TBS 1052B-EDU - 15:26:52 25/02/2022

20cm de distance (capteur-balise)



TBS 1052B-EDU - 15:28:16 25/02/2022

30cm de distance (capteur-balise)

2- Supprimer la composante continue

Pour cela nous utilisons un filtre passe-haut (CR) du premier ordre, car effectivement la composante continue est la fréquence 0Hz. Ici nous prenons une fréquence de coupure de 300Hz, la composante continue est donc supprimée.



4. Chaîne de traitement du signal
Bloc 2 « Supprimer la composante continue »
Etude du LCD I²C et du Bluetooth

Objectifs :

- Concevoir un filtre passif passe-haut d'ordre 1 pour supprimer la composante continue
- Ecrire sur un écran LCD I2C
- Découvrir le Bluetooth

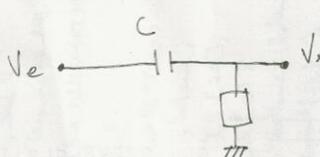
Partie 11 - BLOC 2 - « Supprimer la composante continue »

Dans la vidéo de la chaîne de traitement du signal disponible sur LMS, le bloc numéro 2 est détaillé à partir de la minute 5'46.

Sur labdec, vous devez refaire le montage, et le tester. Bien choisir le signal d'entrée pour le test.

11.1 Schéma du montage pour supprimer la composante continue

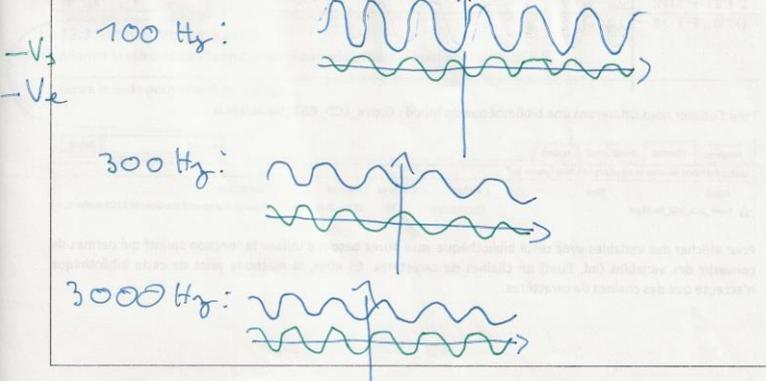
Faire le schéma :



$R = 5000 \Omega$
 $f_c = 300 \text{ Hz}$
 $C = \frac{1}{2\pi R f_c}$

11.2 Test du montage pour supprimer la composante continue

Faire une mini-fiche de mesure ci-dessous :

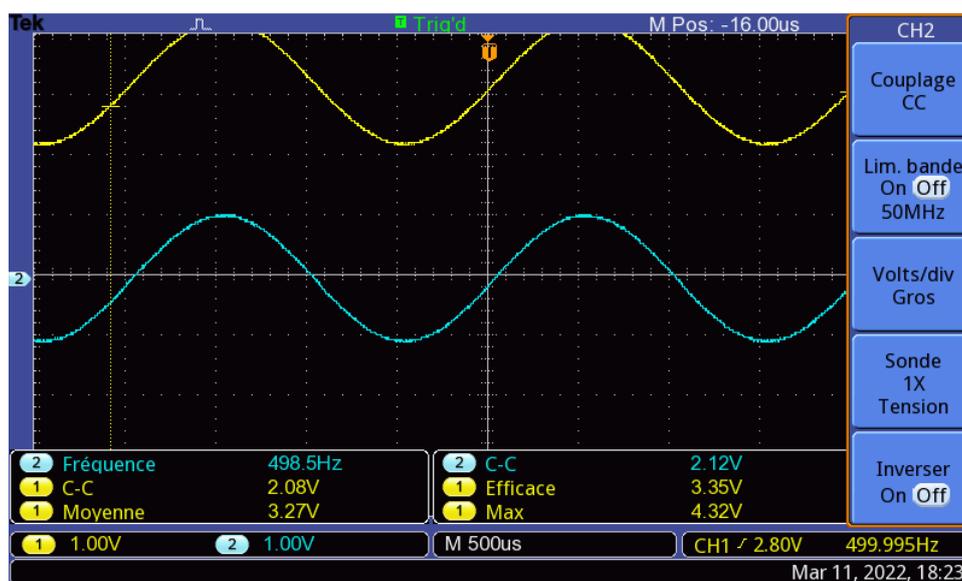


100 Hz:
300 Hz:
3000 Hz:

$-V_s$
 $-V_e$

p. 22

Montage pour supprimer la composante continue



Supprimer la composante continue

3- Amplifier le signal Infrarouge

Nous récupérons donc le signal de sortie du filtre passe-haut pour l'injecter sur la borne (+) d'un TL081 avec la borne (-) reliée à une résistance de 1KOhms reliée à la masse et avec une contre réaction négative munie d'un potentiomètre variant entre 0 et 10kOhms pour avoir un Gain variable.

Il s'agit d'un montage amplificateur non inverseur. La tension de sortie est donc la tension d'entrée amplifiée.

Partie 8 - BLOC 3 - « Amplifier le signal »

Dans la vidéo de la chaîne de traitement du signal disponible sur LMS, le bloc numéro 3 est détaillé à partir de la minute 8'28. Sur labdec, vous devez refaire le montage (Aop TL081 ou TL082), et le tester. Bien choisir le signal d'entrée pour le test.



Fiche de mesure



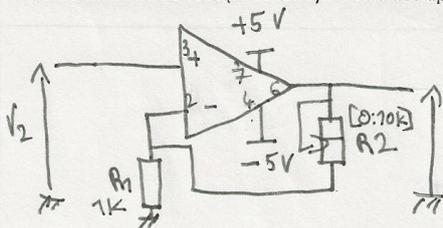
Module ou élément sous test

TL081

Objectif du test

amplifier le signal de sortie du TL081

Schéma du banc de test (avec les symboles des appareils de mesure)



Procédure de test (séquence des opérations)

Le montage a été réalisé, signal d'entrée donné est 5KHz et 500mV d'amplitude.
Nous avons tourné le potentiomètre jusqu'à obtenir une amplification.

Résultats de la mesure (oscillogramme ou tableau de mesures ou caractéristique)

Commentaire des mesures / Conclusion

L'AOP amplifie bien (Malgré la situation)

4- Isoler la fréquence 5000Hz

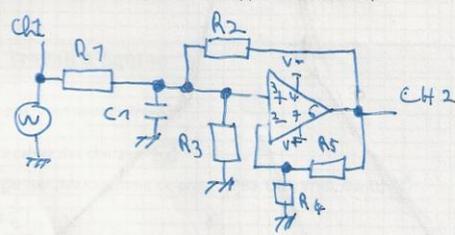
Pour isoler la fréquence de 5kHz nous devons supprimer les fréquences supérieures et inférieures. Pour cela nous utilisons un montage de filtre passe bande (deuxième ordre) avec un facteur de qualité $Q=...$


Fiche de mesure


Module ou élément sous test TL081

Objectif du test isoler la 5 KHz

Schéma du banc de test (avec les symboles des appareils de mesure)



Procédure de test (séquence des opérations)

- Reproduire le schéma
- Regarder ch1 et ch2 à l'oscilloscope

Résultats de la mesure (Tableau de points pour le diagramme de Bode)

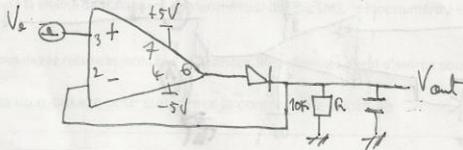
f	100	300	1k	3k	4k	5k	7k	10k	30k
(mV) V_s	5	32	67	270	780	1154	520	176	64
V_e	1	1	1	1	1	1	1	1	1
ΔY	100	95	90	87	73	-58	-80	-85	90
dB	-46	-30,60	-24	-17,58	-2,22	3,46	-70	-75	-24

Commentaire des mesures / Conclusion

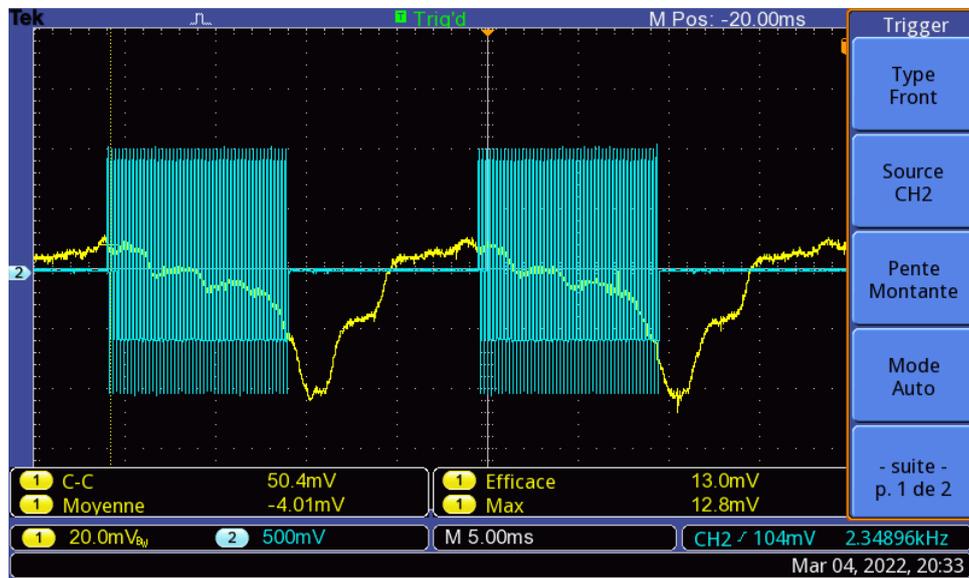
Fiche de mesure TL081 pour isoler la fréquence de 5kHz

5- Récupérer l'enveloppe du signal Infrarouge

Pour cela nous avons créé une diode idéale (sans tension de seuil) à l'aide d'un montage AOp non inverseur et d'une diode. Nous prenons comme signal d'entrée le signal sortant du TL081 permettant d'isoler le 5kHz sur la borne (+) de l'AOp. Nous avons ensuite une résistance de 10kOhms et un condensateur se chargeant lorsque la tension est positive et se déchargeant lorsqu'elle ne l'est pas grâce à la diode idéale.

Partie 9 - BLOC 5 - « Récupérer l'enveloppe »	
Dans la vidéo de la chaîne de traitement du signal disponible sur LMS, le bloc numéro 5 est détaillé à partir de la minute 12'12. Sur labdec, vous devez refaire le montage (AOP TL081 ou TL082), et le tester. Bien choisir le signal d'entrée pour le test.	
 Fiche de mesure 	
Module ou élément sous test TL081	
Objectif du test Récupérer l'enveloppe	
Schéma du banc de test (avec les symboles des appareils de mesure) 	
Procédure de test (séquence des opérations) $R = 10\text{K}\Omega$ et $C = \frac{0,002}{10\text{kHz}} = 20\text{ nF}$	
Résultats de la mesure (oscillogramme ou tableau de mesures ou caractéristique)  1M4748	
Commentaire des mesures / Conclusion RC trop petit.	

Fiche de mesure TL081 Pour récupérer l'enveloppe du signal



TBS 1052B-EDU - 14:53:17 04/03/2022

Récupérer l'enveloppe

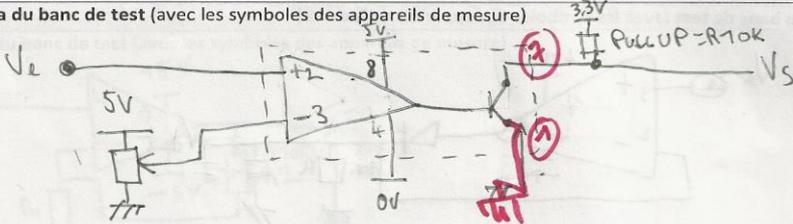
6- Rendre le signal binaire

Rendre le signal binaire permet de communiquer avec le microcontrôleur (FRDM-KL25Z). Pour cela nous utilisons l'AOp LM311 avec en signal d'entrée le signal de sortie précédent (V5) sur la borne (+) et une référence de tension avec un potentiomètre branché entre le 5V et le 0V sur la borne (-).

En sortie nous avons un transistor qui s'active si V5 est supérieur à la référence de tension. Et l'on va générer une tension (V6) grâce à une résistance de Pull 'Up connecté au 3.3V. Si le transistor est ouvert nous avons 0V, si le transistor est fermé nous avons 3.3V.

Partie 10 - BLOC 6 - « Rendre le signal binaire »

Dans la vidéo de la chaîne de traitement du signal disponible sur LMS, le bloc numéro 6 est détaillé à partir de la minute 17'04. Sur labdec, vous devez refaire le montage, et le tester. Bien choisir le signal d'entrée pour le test.

	Fiche de mesure	
Module ou élément sous test LM311		
Objectif du test Rendre le signal binaire		
Schéma du banc de test (avec les symboles des appareils de mesure) 		
Procédure de test (séquence des opérations) <ul style="list-style-type: none">- Branchement et câblage de l'ACP- Vérifier que le signal est bien binaire- Vérifier que faire varier le potentiomètre change bien V_s de référence.		
Résultats de la mesure (oscillogramme ou tableau de mesures ou caractéristique)		
Commentaire des mesures / Conclusion Nous obtenons bien un signal binaire dont le niveau de référence est réglé avec un potentiomètre.		

Fiche de mesure LM311 Pour rendre le signal binaire

Carte Mezzanine

Après avoir terminé l'étude de la carte mezzanine, il nous a fallu la concevoir, l'imprimer, la souder, puis la tester, pour enfin la monter sur le robot pour pouvoir l'employer à son usage initialement prévu. C'est à dire accueillir 2 capteurs ultrason de distance, un connecteur Groove pour y ajouter un LCD. Et surtout toute la chaîne de traitement du signal Infrarouge afin de capter la direction de la balise pour la rejoindre (+ la création du +5V/-5V pour alimenter les AOPs de la chaîne).

a) Préparation

Premièrement, Adrien a réalisé le schéma sur feuille blanche afin de réunir toutes les parties de l'étude en un seul et même schéma. Cela nous a permis d'établir ensuite la liste des composants (car ils sont nombreux sur cette carte).

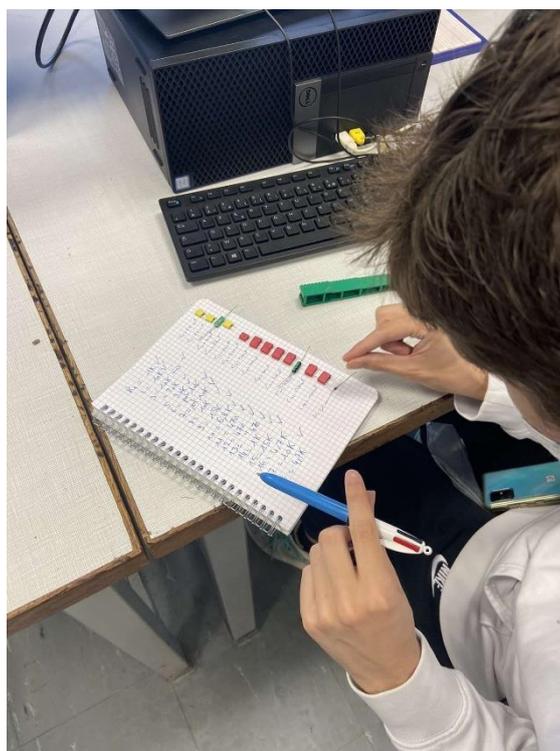
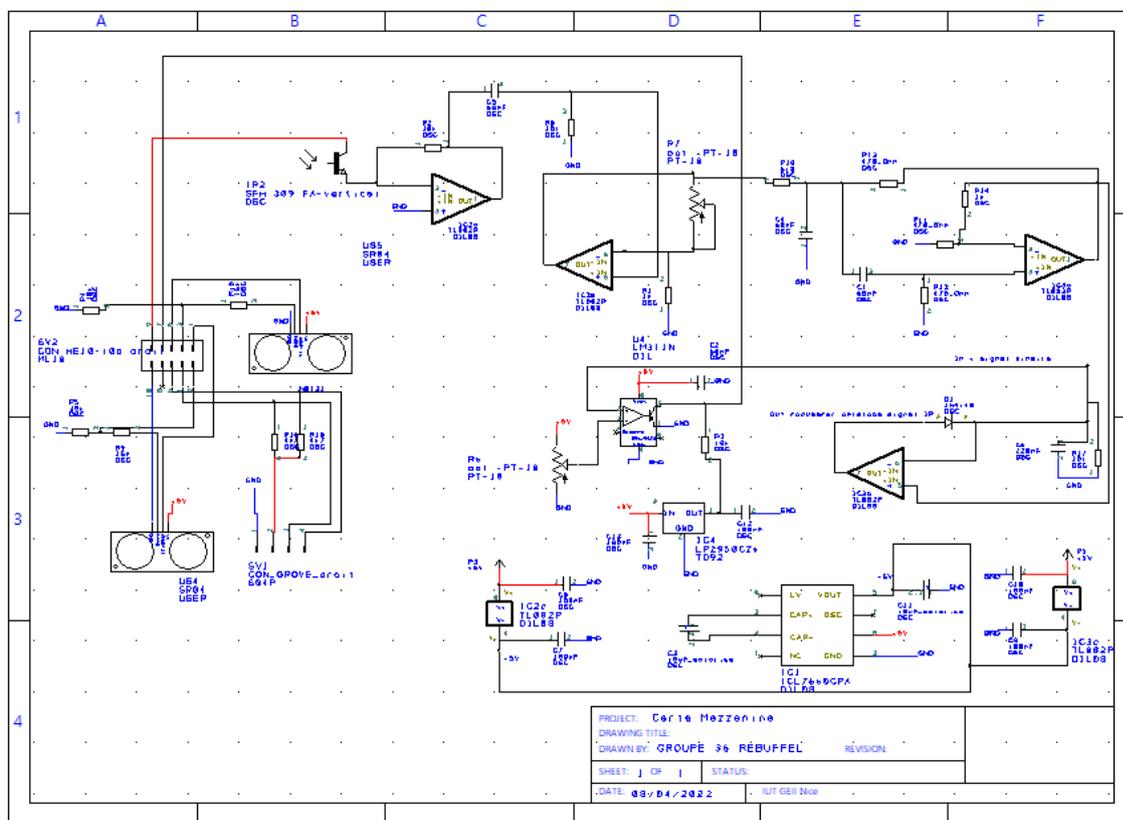


Photo de Adrien qui établit la liste des composants

b) Conception du schématique

Après avoir réalisé cette tâche à la fois fastidieuse mais primordiale, nous avons pu commencer la conception du schématique de notre carte sur le logiciel DesignSpark. Notre groupe de quatre étudiants étant totalement novice dans la conception de carte sur logiciel, nous avons commencé "petit à petit" et avec plusieurs membres du groupe en même temps pour être sûr de ne pas commettre d'erreur (qui pourrait être fatal sur le fonctionnement du robot et aussi sur notre note).



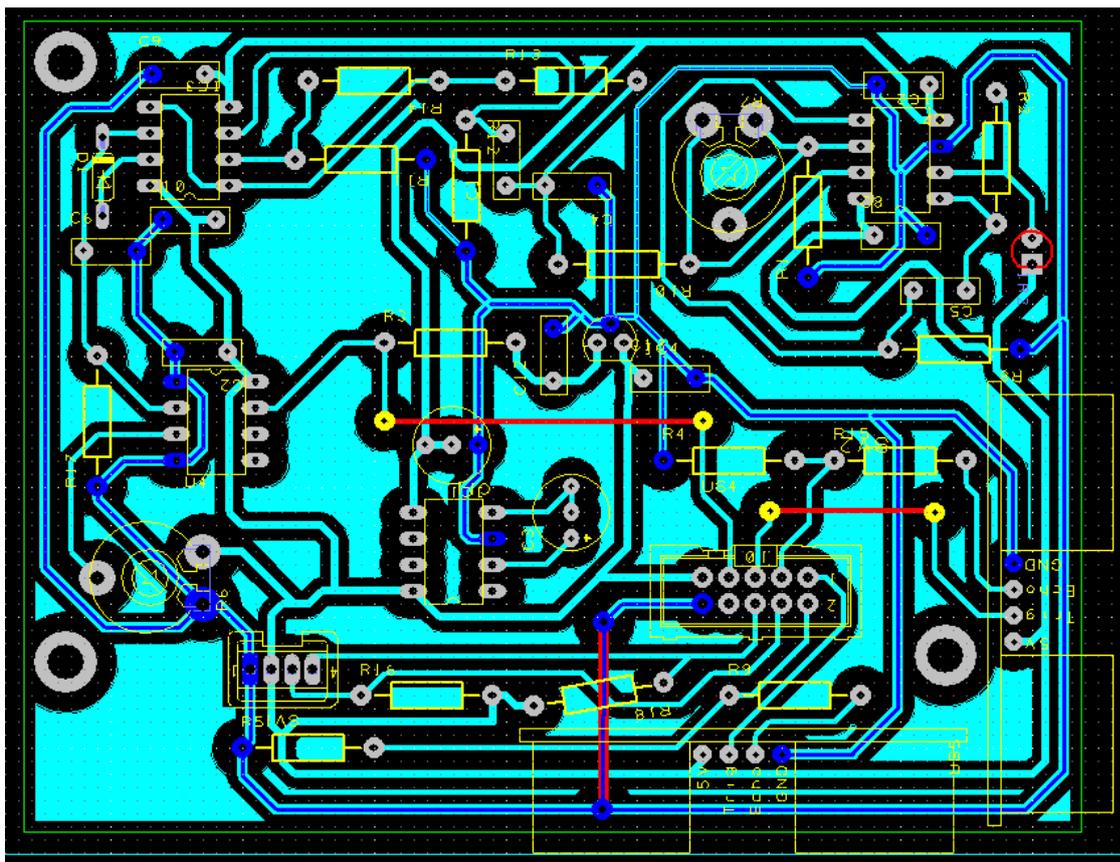
Capture d'écran du schématique

c) Conception du PCB

Dès la fin de la réalisation et la validation du schématique, nous nous sommes rendu compte que celui-ci était assez mal organisé et difficile à comprendre pour une personne qui n'as pas participé à sa conception.

Les points à améliorer pour de futurs projets sont notamment :
Occuper de façon homogène l'espace disponible dans le cartouche ;
Organiser les différentes parties de la carte dans des blocs nommés puis, Placer tous les composants dans le bon sens pour rendre plus lisible le schéma et un éventuel retour sur cette conception.

Après validation de celui-ci, nous avons pu réaliser la version PCB, ce qui est la dernière étape avant l'impression finale de la carte.



Capture d'écran du PCB

La conception du PCB s'est faite en suivant un cahier des charges bien précis imposé par les professeurs.

Les consignes étaient les suivantes :

1. La taille maximum de la carte de 10x10 cm
2. Le nombre de straps limité à 3
3. La distance entre les trous de fixation doit être écarté de façon à pouvoir fixer la carte par-dessus la carte commande (d'où le nom "carte mezzanine").

Ce travail était assez long (De l'ordre de plusieurs demi-journées) mais en appliquant une concentration suffisante nous avons pu respecter à la fois les consignes du cahier des charges et les possibilités techniques.

d) Impression et soudures

Par la suite, la prochaine étape était l'impression de la carte. Nous avons réalisé cette tâche en autonomie à l'aide de la procédure de tirage PCB disponible sur LMS.



Photo de la carte mezzanine imprimée

Suite à ça, la dernière étape était de souder les composants et de tester chaque connexion afin de s'assurer du bon fonctionnement de la carte avant de la brancher au reste du robot (pour éviter de griller et gaspiller le fusible ou un composant).

e) Problèmes rencontrés

Nous avons fait face à trois problèmes majeurs :

Le premier était causé par l'impression de la carte. Plusieurs minuscules lignes de cuivre ne s'étaient pas correctement fait supprimer par le graveur laser. Il nous a fallu gratter avec une pointe en métal pour faire disparaître l'excédent de cuivre.

Le second était une erreur de conception au niveau de la chaîne de traitement. Nous avons trouvé grâce au test de continuité que le problème venait de la partie filtrage de la chaîne. Pour que le traitement du signal fonctionne correctement, il utilise un filtre passe-bande de Sallen Key. Malheureusement, une erreur* dans la simulation de ce filtre nous a fait choisir de mauvais composants (3 résistances mal dimensionné au total). Nous avons donc effectué le changement de ces trois résistances.

*Explication de l'erreur : sur le logiciel Filter Pro (utilisé pour simuler les filtres), on avait choisi un facteur de qualité $Q = 50$. Or l'AOP TL082 qu'on utilisait pour ce filtre ne supporte pas un facteur de qualité aussi important. On a donc refait sur le logiciel le filtre avec cette fois-ci un facteur de qualité $Q = 4$, ce qui nous donne des valeurs différentes pour 3 résistances du filtre.

Et enfin, le dernier problème qui empêchait le bon fonctionnement de cette carte était lié aux broches SDA/SCL du Mbed. Pendant la conception de la carte, nous n'avons pas fait attention de relier les bornes du connecteur Groove pour l'écran LCD, aux bonnes bornes du connecteur HE10 qui est directement relié au mbed. Pour réparer cette erreur nous avons déplacé les 2 pistes en question à l'aide de fils soudé sous la carte.

Après toutes ces étapes importantes, nous avons pu passer au code de chaque épreuve.

Epreuves

a) Suivre une ligne blanche

Premièrement, nous nous sommes occupés de l'épreuve du suivi de ligne, c'est une épreuve que nous connaissons car nous avons à la réaliser au premier semestre. La méthode est simple, nous avons 4 capteurs de ligne blanche en dessous du robot. Les deux capteurs du milieu qui servent à obtenir un coefficient d'éloignement (sois trop à droite sois trop à gauche), ce qui assure un suivi presque parfait de la ligne. Puis les deux capteurs aux extrémités du robot, qui servent en dernier recours pour engager un virage fort pour retrouver la ligne (Voir le code dans l'annexe).

b) Sortir du labyrinthe

Pour l'épreuves du labyrinthe, le besoin de réfléchir à une méthode pour en sortir s'est fait sentir. Après quelques minutes de réflexions nous avons choisis de coller le mur droit du labyrinthe, ce qui nous mènerais forcément à la sortie. Pour ce faire, nous avons mis 2 capteurs de distance à ultrason, l'un vers l'avant du robot et l'autre vers la droite pour détecter les murs.

- Pour tourner à droite il fallait que le capteur droit ne capte plus de mur.
- Pour tourner à gauche il fallait que le capteur de l'avant capte un mur.
- Pour aller tout droit le robot doit suivre le mur droit.

Nous nous sommes aussi rendu compte que le robot n'avançait jamais en ligne bien droite. Pour pailler à ce problème, nous avons rajouter des petites corrections dans le code pour que le robot soit toujours à une bonne distance du mur. (Voir le code dans l'annexe)

Voici un schéma représentant le labyrinthe à effectuer pendant l'épreuve d'essai robot 2 :

c) Rejoindre la balise

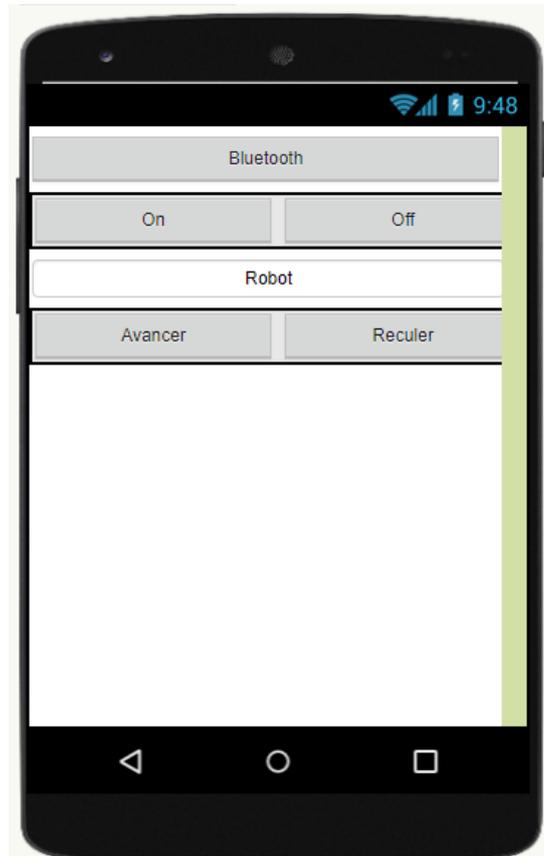
Le programme de la balise était, selon nous, le plus simple à réaliser. En effet la machine à l'état de ce programme n'avait que trois étapes. La première servait à chercher la balise (le robot tourne sur lui-même jusqu'à trouver la balise) La seconde étape sert à avancer vers la balise. Puis la troisième, à s'arrêter dès que les capteurs étaient sur du blanc c'est à dire au pied de la balise (Voir le code dans l'annexe).

Télécommande Bluetooth

Afin de pouvoir télécommander le robot avec mon téléphone, j'ai utilisé une carte Bluetooth (HC-05) , l'application « Bluetooth Serial Controller » et le logiciel « app-inventor ».

Dans la première partie je devais alimenter ma carte Bluetooth pour sa on a dû souder des fils male femelle sur la carte commande puis nous avons fait la liaison RX TX, entre la carte Bluetooth et ma Mbed. On réussit à envoyer des caractères avec l'application Bluetooth Serial Controller et recevoir avec ma mbed puis on récupère se caractère avec mon code cc pour faire au robot ce que je voulais (par exemple avancer, reculer, tourner à droite ect...)

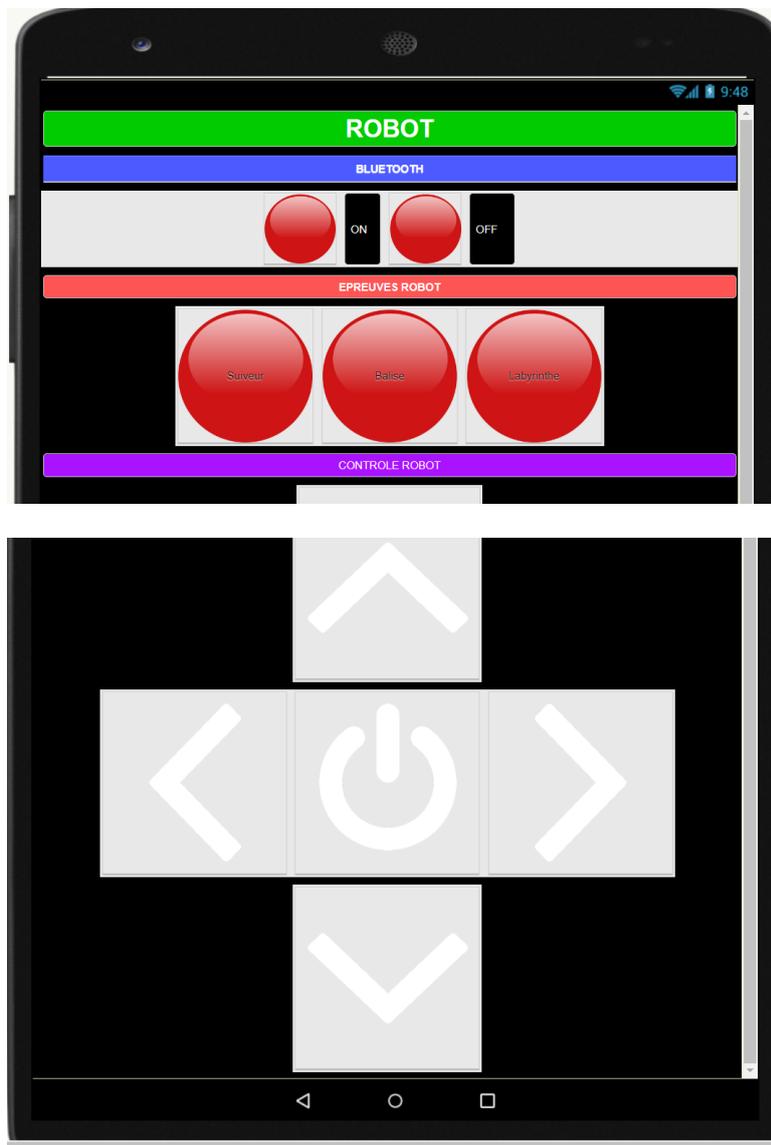
Puis j'ai designé une application sur le logiciel app inventor.



Premier design d'application :

Comme nous pouvons le voir dans la photo j'ai mis l'application en format tablette car j'avais ma tablette du lycée qui ma servit de support. J'ai réussi à mettre des boutons pour contrôler le robot (avancer, reculer, droite et gauche). Je ne me suis pas arrêté là, enfaite je me suis servi de l'application comme une IHM j'ai créé des boutons qui permettent de choisir l'épreuve voulu.

Design final :



(Voir le code dans l'annexe)

L'application permet de :

- Contrôler le robot (droite gauche devant et arrière)
- Choisir l'épreuve voulu

Conclusion/Remarques

Pour conclure, nous avons réussi à respecter les échéances prévues dans le diagramme de Gantt, à l'exception du test de la carte mezzanine. En effet les problèmes que nous avons rencontrés nous ont fortement retardé. Malgré cela, nous avons réussi à respecter le cahier des charges et à compléter toutes les épreuves.

Durant ce projet nous avons appris à concevoir le schématique et le PCB d'une carte, à développer le code des épreuves et pour finir, à travailler en équipe et à organiser un projet comme celui-ci. Finalement nous sommes unanimes pour dire que nous avons correctement organisé ce projet.

Quant à la décoration (qui était déjà présente sur le robot) est inspirée de Flash McQueen de la saga Cars que l'on vous conseille fortement !

Annexes

Screenshot du code de l'épreuve du Labyrinthe

```
1 #include "mbed.h"
2 #include "SRF05.h"
3 #include "Grove_LCD_RGB_Backlight.h"
4 #define T 0.0001
5
6 AnalogIn batterie(A0);
7
8 PwmOut MotD(D6);
9 PwmOut MotG(D8);
10 DigitalOut sensD(D9);
11 DigitalOut sensG(D7);
12 SRF05 US5(D5,D10);
13 SRF05 US4(D11,D12);
14 Grove_LCD_RGB_Backlight lcd(D14,D15);
15 AnalogIn CapG(A4);
16 AnalogIn CapD(A1);
17 AnalogIn CapED(A2);
18 AnalogIn CapEG(A3);
19
20 char tab[]="AVANCE          ";
21 char tabd[]="CORRIGE A DROITE";
22 char tabg[]="CORRIGE A GAUCHE";
23 char tabtd[]="TOURNE A DROITE";
24 char tabtg[]="TOURNE A GAUCHE";
25
26 int main()
27 {
28     float ValCapD,ValCapG,E,ValCapEG,ValCapED;
29     lcd.setRGB(255,0,0);
30     MotD.period(T);
31     MotG.period(T);
32     float f1,f2;
33     wait(3);
34     while(1) {
```

```

35 ValCapED=CapED.read();
36 ValCapEG=CapEG.read();
37 ValCapD=CapD.read();
38 ValCapG=CapG.read();
39 lcd.locate(0,0);
40 lcd.print(tab);
41 sensG.write(0); //moteur on
42 sensD.write(0);
43 MotD.pulsewidth_us(30);
44 MotG.pulsewidth_us(33.8);
45 f1 = US5.read();
46 f2 = US4.read();
47 printf("Measured : %.1f et %.1f \n\r", f1,f2);
48
49 if ((f2>25)&&(f2<35)) { //tourner a droite UNPEU
50     puts("droite unpeu");
51     lcd.locate(0,0);
52     lcd.print(tabd);
53     sensG.write(1);
54     sensD.write(0);
55     MotD.pulsewidth_us(0);
56     MotG.pulsewidth_us(50);
57     wait(0.1);
58     sensG.write(0);
59     sensD.write(0);
60     MotD.pulsewidth_us(0);
61     MotG.pulsewidth_us(0);
62     wait(1);
63 }
64
65 if (f2>35) { //tourner a droite BCP
66     puts("droite bcp");
67     lcd.locate(0,0);

```

```

68     lcd.print(tabtd);|
69     wait(1);
70     sensG.write(0);
71     sensD.write(0);
72     MotD.pulsewidth_us(0);
73     MotG.pulsewidth_us(50);
74     wait(0.4);
75     sensG.write(0);
76     sensD.write(0);
77     MotD.pulsewidth_us(0);
78     MotG.pulsewidth_us(0);
79     wait(1);
80 } else if ( f1<15) { //tourner a gauche
81     puts("gauche");
82     lcd.locate(0,0);
83     lcd.print(tabtg);
84     sensG.write(0);
85     sensD.write(1);
86     MotD.pulsewidth_us(50);
87     MotG.pulsewidth_us(50);
88     wait(0.35);
89     sensG.write(0);
90     sensD.write(0);
91     MotD.pulsewidth_us(0);
92     MotG.pulsewidth_us(0);
93     wait(0.7);
94 }
95 if ((f2<15)) { //tourner a gauche UNPEU
96     lcd.locate(0,0);
97     lcd.print(tabg);
98     sensG.write(0);
99     sensD.write(1);
100     MotD.pulsewidth_us(50);
101     MotG.pulsewidth_us(100);
102     wait(0.1);
103     sensG.write(0);
104     sensD.write(0);
105     MotD.pulsewidth_us(0);
106     MotG.pulsewidth_us(0);
107     wait(1);
108     if ((ValCapD<0.3) && (ValCapG<0.3));
109         sensG.write(0);
110         sensD.write(0);
111         MotD.pulsewidth_us(0);
112         MotG.pulsewidth_us(0);
113
114     }
115 }
116 }

```

Screenshot du code de l'épreuve du Suivi de ligne à 25cm d'un obstacle

```

1 #include "mbed.h"
2 #include "SRF05.h"
3 #define periode 100
4
5 PwmOut mot1(D6);
6 PwmOut mot2(D8);
7 AnalogIn CapG(A4);
8 AnalogIn CapD(A1);
9 DigitalIn FDC(D2);
10 DigitalIn Jack(D3);
11 DigitalOut sensG(D7);
12 DigitalOut sensD(D9);
13 SRF05 US5(D5,D10);
14 AnalogIn CapED(A2);
15 AnalogIn CapEG(A3);
16
17
18
19 void vitesse(int, int);
20
21 int main()
22 {
23     float ValCapD,ValCapG,E,ValCapEG,ValCapED;
24     mot1.period_us(periode);
25     mot2.period_us(periode);
26     mot1.pulsewidth_us(periode);
27     mot2.pulsewidth_us(periode);
28     float E;
29     float MD,MG,f1;
30     int ValJack,ValFDC,etat=0,K;
31     while(1)
32     {
33         f1 = US5.read();
34

```

```

35     ValCapD=CapD.read();
36     ValCapG=CapG.read();
37     ValJack=Jack.read();
38     ValFDC=FDC.read();
39
40     ValCapED=CapED.read();
41     ValCapEG=CapEG.read();
42     ValCapD=CapD.read();
43     ValCapG=CapG.read();
44     E=ValCapG-ValCapD;
45     K=35;
46     MG=28-E*K;
47     MD=35+E*K;
48     //printf("etat=%d g=%.2f d=%.2f err=%.2f MG=%.0f MD=%.0f \n\r",etat,ValCapG,ValCapD,E,MG,MD);
49     //printf("etat=%d dist=%f \n \r", etat, f1);
50     switch(etat)
51     {
52     case 0:
53         if(ValJack==1)
54             etat=1;
55             break;
56     case 1:
57         if(ValFDC==0)
58             etat=2;
59         if (f1<=25)
60             etat=3;
61             break;
62     case 2:
63         if(ValJack==0)
64             etat=0;
65             break;
66
67         case 3:
68             if (f1>=25)
69                 etat=1;
70                 break;
71     }
72     switch(etat)
73     {
74     case 0:
75         vitesse(0, 0);
76         break;
77     case 1:
78         vitesse(MG, MD);
79         break;
80     case 2:
81         vitesse(0, 0);
82         break;
83
84     case 3:
85         vitesse (0, 0);
86         break;
87     }
88     }
89 }
90
91
92 void vitesse(int gauche, int droit)
93 {
94     if (gauche>=0) {
95         sensG.write(0);
96         mot1.pulsewidth_us(gauche);
97     }

```

```
98     if (droit>=0) {
99         sensD.write(0);
100         mot2.pulsewidth_us(droit);
101     }
102
103     if (gauche<0) {
104         sensG.write(1);
105         mot1.pulsewidth_us(periode+gauche);
106     }
107     if (droit<0) {
108         sensD.write(1);
109         mot2.pulsewidth_us(periode+droit);
110     }
111 }
```

Screenshot du code de l'épreuve du Suivi de Balise

```

1 #include "mbed.h"
2 #define T 0.0001
3
4 DigitalIn BAL(D4);
5 PwmOut MotD(D6);
6 PwmOut MotG(D8);
7 DigitalOut sensD(D9);
8 DigitalOut sensG(D7);
9 DigitalIn FDC(D2);
10 AnalogIn CapG(A4);
11 AnalogIn CapD(A1);
12 AnalogIn CapED(A2);
13 AnalogIn CapEG(A3);
14 float ValCapD,ValCapG,E,ValCapEG,ValCapED;
15
16 int main()
17 {
18     MotD.period(T);
19     MotG.period(T);
20     int bal,etat=0,ValFDC;
21     while(1) {
22         ValCapED=CapED.read();
23         ValCapEG=CapEG.read();
24         ValCapD=CapD.read();
25         ValCapG=CapG.read();
26         bal=BAL.read();
27         ValFDC=FDC.read();
28         //printf("%d %d\n",etat,ValFDC);
29         //printf("VG=%f VD=%.2f VEG=%.2f VED=%.2f  etat=%d\n\r",ValCapG,ValCapD,ValCapEG,ValCapED,etat);
30         switch(etat) {
31             case 0:
32                 if (bal==1) etat=1;
33                 if ((ValCapD<0.4)&& (ValCapG<0.4)) etat=10;
34                 break;

```

```

35
36     case 1:
37         if (bal==0) etat=0;
38         if ((ValCapD<0.3)&& (ValCapG<0.3)) etat=10;
39         break;
40     }
41     switch(etat) {
42     case 0:
43         sensG.write(1);
44         sensD.write(0);
45         MotD.pulsewidth_us(75);
46         MotG.pulsewidth_us(25);
47         break;
48
49
50     case 1:
51         sensG.write(0);
52         sensD.write(0);
53         MotD.pulsewidth_us(25);
54         MotG.pulsewidth_us(25);
55         break;
56
57     case 10:
58         sensG.write(0);
59         sensD.write(0);
60         MotD.pulsewidth_us(0);
61         MotG.pulsewidth_us(0);
62         break;
63     }
64 }
65 }
66

```

Screenshot du code de la télécommande Bluetooth

1) partie mbed

```

1 #include "mbed.h"
2 #define T 0.0001
3
4 DigitalOut ledV(D4);
5 DigitalOut ledJ(D5);
6 DigitalOut ledR(D6);
7 DigitalIn bp1(D12);
8 DigitalIn bp2(D13);
9 DigitalIn bp3(D14);
10 PwmOut MotD(D6);
11 PwmOut MotG(D8);
12 DigitalOut sensD(D9);
13 DigitalOut sensG(D7);
14
15 Ticker tick1;
16 Ticker tick2;
17
18 Serial test(PTE22,PTE23);
19
20 void envoiEtat();
21 void lectureBluetooth();
22
23 int main()
24 {
25     test.baud(9600);
26     MotD.period(T);
27     MotG.period(T);
28
29     tick1.attach(&envoiEtat,0.1);
30
31
32     while(1) {
33         printf("ON/FF = %d Droite = %d Gauche = %d\n",ledV.read(), ledJ.read(),ledR.read());
34     }

```

```

35 }
36
37
38 void envoiEtat()
39 {
40     if(bp1.read() == 0) {
41         test.printf("a");
42     } else {
43         test.printf("d");
44     }
45     if(bp2.read() == 0) {
46         test.printf("g");
47     } else {
48         test.printf("r");
49     }
50     if(bp3.read() == 0) {
51         test.printf("l");
52     } else {
53         test.printf("s");
54     }
55
56     if(test.readable()) {
57         char message;
58         message = test.getc();
59         if(message == 'a') {
60             ledV = !ledV;
61             sensG.write(0);
62             sensD.write(0);
63             MotD.pulsewidth_us(50);
64             MotG.pulsewidth_us(54);
65             wait(0.5);
66             MotD.pulsewidth_us(0);
67             MotG.pulsewidth_us(0);

```

```

68     }
69     if(message == 'g') {
70         ledJ = !ledJ;
71         sensG.write(0);
72         sensD.write(1);
73
74         MotD.pulsewidth_us(50);
75
76         MotG.pulsewidth_us(50);
77         wait(0.3);
78         sensG.write(0);
79         sensD.write(0);
80         MotD.pulsewidth_us(0);
81         MotG.pulsewidth_us(0);
82
83     }
84     if(message == 'd') {
85         ledR = !ledR;
86         sensG.write(1);
87         sensD.write(0);
88         MotD.pulsewidth_us(50);
89         MotG.pulsewidth_us(50);
90         wait(0.3);
91         sensG.write(0);
92         sensD.write(0);
93         MotD.pulsewidth_us(0);
94         MotG.pulsewidth_us(0);
95     }
96     if(message == 'r') {
97         ledV = !ledV;
98         sensG.write(1);
99         sensD.write(1);
100        MotD.pulsewidth_us(50);
101
102        MotG.pulsewidth_us(46);
103        wait(0.5);
104        sensG.write(0);
105        sensD.write(0);
106        MotD.pulsewidth_us(0);
107        MotG.pulsewidth_us(0);
108    }
109    if(message == 'l') {
110        printf("labbbbb");
111    }
112    if(message == 's') {
113        printf("suiveur");
114    }
115    }
116

```

2)partie app inventor

```
when ListPicker1 .BeforePicking
do set ListPicker1 .Elements to BluetoothClient1 .AddressesAndNames
```

```
when ListPicker1 .AfterPicking
do set ListPicker1 .Selection to call BluetoothClient1 .Connect
address ListPicker1 .Selection
set ListPicker1 .Text to "Connect"
```

```
when Button17 .Click
do call BluetoothClient1 .SendText
text "1"
```

```
when Button18 .Click
do call BluetoothClient1 .SendText
text "0"
```

```
when Button19 .Click
do call BluetoothClient1 .SendText
text "a"
```

```
when Button23 .Click
do call BluetoothClient1 .SendText
text "d"
```

```
when Button20 .Click
do call BluetoothClient1 .SendText
text "g"
```

```
when Button21 .Click
do call BluetoothClient1 .SendText
text "r"
```

```
when Button14 .Click
do call BluetoothClient1 .SendText
  text " s "

when Button15 .Click
do call BluetoothClient1 .SendText
  text " l "

when Button16 .Click
do call BluetoothClient1 .SendText
  text " b "
```